SENSOR LETTERS
Vol. 6, 1–13, 2008

A Cellular Learning Automata Based Clustering Algorithm for Wireless Sensor Networks

M. Esnaashari[1,*], and M. R. Meybodi[1,2]

[1] *Soft Computing Laboratory, Computer Engineering and Information Technology Department, Amirkabir University of Technology, Tehran, Iran*
[2] *Institutes for Studies in Theoretical Physics and Mathematics (IPM), School of Computer Science, Tehran, Iran*

(Received: xx Xxxx xxxx. Revised/Accepted: xx Xxxx xxxx)

In the first part of this paper, we propose a generalization of cellular learning automata (CLA) called irregular cellular learning automata (ICLA) which removes the restriction of rectangular grid structure in traditional CLA. In the second part of the paper, based on the proposed model a new clustering algorithm for sensor networks is designed. The proposed clustering algorithm is fully distributed and the nodes in the network don't need to be fully synchronized with each other. The proposed clustering algorithm consists of two phases; initial clustering and reclustering. Unlike existing methods in which the reclustering phase is performed periodically on the entire network, reclustering phase in the proposed method is performed locally whenever it is needed. This results in a reduction in the consumed energy for reclustering phase and also allows reclustering phase to be performed as the network operates. The proposed clustering method in comparison to existing methods produces a clustering in which each cluster has higher number of nodes and higher residual energy for the cluster head. Local reclustering, higher residual energy in cluster heads and higher number of nodes in each cluster results in a network with longer lifetime. To evaluate the performance of the proposed algorithm several experiments have been conducted. The results of experiments have shown that the proposed clustering algorithm outperforms existing clustering methods in terms of quality of clustering measured by the total number of clusters, the number of sparse clusters and the remaining energy level of the cluster heads. Experiments have also shown that the proposed clustering algorithm in comparison to other existing methods prolongs the network lifetime.

Keywords: Sensor Networks, Clustering, Learning Automata, Cellular Learning Automata.

RESEARCH ARTICLE

1. INTRODUCTION

Cellular learning automata (CLA) which is introduced recently,[1,48,47] is a powerful mathematical model for many decentralized problems and phenomena. CLA is obtained by combining cellular automata (CA) and learning automata (LA). The basic idea of CLA is to utilize learning automata to adjust the state transition probability of stochastic CA. A CLA is a CA in which one learning automaton is assigned to every cell. This model is superior to CA because of its ability to learn and also is superior to a single learning automaton because it is a collection of learning automata, which can interact with each other and solve a particular problem. CLA has found many applications such as image processing,[6–9] rumor diffusion,[10] modeling of commerce networks,[8] channel assignment in cellular networks[11] and VLSI placement,[12] to mention a few.

A wireless sensor network consists of many sensor nodes. The sensors which are randomly deployed in the environment of a phenomenon play the role of gathering specific data from the environment, processing and finally sending it to the base station (sink). Sensor networks have critical applications in the scientific, medical, commercial, and military domains. Wireless sensor networks can be exposed to highly dynamic and hostile environments,[50] and therefore, they must be tolerant to the failure and loss of connectivity of individual nodes. Clearly, algorithms for wireless sensor networks must be distributed in order to prevent single points of failure. Moreover, sensors in the network are limited in power, computational capacity and

*Corresponding author; E-mail:

memory, and they have only short-range radio transmission. Therefore in a given network, the sensor nodes must be intelligent enough to make the network capable to self-organizing and self-healing.

The major design goal in a power-aware communication protocol for wireless sensor networks is to minimize energy consumption and maximize the system lifetime. Clustering is one of the approaches for achieving such design goal. The main idea of the clustering is based on dynamically electing a cluster head among eligible active nodes. In the clustering process, some of the sensor nodes with a defined radio range make a cluster together choosing a node as the cluster head (hereafter called head) whose responsibility is to communicate with the heads of other clusters. It is possible that in the future these clusters are combined to create a bigger cluster. Usually, at the end of a clustering algorithm, the nodes are organized into disjoint sets (clusters). Each cluster consists of a head which manages the cluster and some followers (hereafter called members) which are usually located within the communication radius of the head. Each node is a member of only one cluster. The purpose of a clustering algorithm is to find a suitable clustering with proper number of clusters which cover the whole network.

Clustering sensor nodes has some advantages such as data aggregation in order to reduce energy consumption by sensors,[15,42] facilitates queries on the sensor network,[43] form an infrastructure for scalable routing,[28,51] and efficient network-wide broadcast.[52] Clustering algorithms for sensor networks fall into two categories, centralized clustering and distributed clustering.[53] Centralized clustering protocols require the global network knowledge, introduce substantial storage, communication and computation overheads and thus are not desirable for resource-constrained sensor nodes. Distributed clustering algorithms usually make decisions based on localized information.[54–56] In general, distributed clustering schemes introduce less communication cost when compared with centralized schemes.

Clustering algorithms proposed in the literature usually consist of two phases; initial clustering phase and reclustering phase.[15,27,45] Initial clustering is performed at the network startup resulting in an infrastructure in which each node of the network is either a head or a member. Using this infrastructure, the normal operation of the network is started. In many real scenarios, the normal operation of the network is a simple data gathering scheme which is performed as follows: members periodically gather data from the environment, process the data and then send it out to the heads. Heads periodically aggregate data received from the members of their clusters and forward aggregated data towards the sink using a multi-hop inter-cluster routing scheme. Heads consume lots of energy in such a data gathering scenario because of two main reasons; Receiving data from all of their members and inter-cluster communications which need higher transmission range than intra cluster communications. Thus, to prevent heads from rapid energy exhaustion, reclustering is performed in the network from time to time to change the role of the nodes as heads or members.

In the first part of this paper, we propose a generalization of cellular learning automata (CLA) called irregular cellular learning automata (ICLA) which removes the restriction of rectangular grid structure in traditional CLA. In the second part of the paper, based on the proposed model we design a clustering algorithm for wireless sensor networks. The proposed clustering algorithm is fully distributed and the nodes do not need to be fully synchronized with each other. The proposed clustering algorithm consists of two phases; initial clustering and reclustering. Unlike static reclustering schemes, the reclustering scheme proposed in this paper is performed locally and adaptively whenever it is needed. A local reclustering is only required if a head consumes so much energy that cannot continue its role as a head. In such a situation, a local reclustering is initiated in the cluster managed by that head. The role of the head is transferred to the newly elected head(s) at the end of the local reclustering phase, thus normal operation of the network need not to be stopped during the reclustering phase. Local reclustering reduces the amount of energy consumed for changing the infrastructure of the network. The proposed clustering method in comparison to existing methods produces a clustering in which each cluster has higher number of nodes and higher residual energy for the cluster heads. Local reclustering, higher residual energy in cluster heads and higher number of nodes in each cluster results in a network with longer lifetime. To evaluate the performance of the proposed algorithm several experiments have been conducted. The results of experiments have shown that the proposed clustering algorithm outperforms existing clustering methods in terms of quality of clustering measured by the total number of clusters, the number of sparse clusters and the remaining energy level of the cluster heads. Experiments have also shown that the proposed clustering algorithm in comparison to other existing methods prolongs the network lifetime.

The rest of this paper is organized as follows. In Section 2 cellular automata, learning automata and cellular learning automata are briefly reviewed and then irregular cellular learning automata is introduced. Section 3 gives an overview on clustering algorithms for wireless sensor networks. The proposed ICLA based clustering algorithm is described in Section 4. Simulation results are given in Section 5. Section 6 is the conclusion.

2. CELLULAR LEARNING AUTOMATA

In this section we briefly review cellular automata, learning automata, cellular learning automata and then introduce irregular cellular learning automata.

2.1. Cellular Automata

Cellular automata are mathematical models for systems consisting of large number of simple identical components with local interactions. CA is a non-linear dynamical system in which space and time are discrete. It is called cellular because it is made up of cells like points in a lattice or like squares of checker boards, and it is called automata because it follows a simple rule.[2] The simple components act together to produce complicated patterns of behavior. Cellular automata perform complex computations with a high degree of efficiency and robustness. They are especially suitable for modeling natural systems that can be described as massive collections of simple objects interacting locally with each other.[3,4] Informally, a d-dimensional CA consists of an infinite d-dimensional lattice of identical cells. Each cell can assume a state from a finite set of states. The cells update their states synchronously on discrete steps according to a local rule. The new state of each cell depends on the previous states of a set of cells, including the cell itself, and constitutes its neighborhood.[5] The state of all cells in the lattice is described by a configuration. A configuration can be described as the state of the whole lattice. The rule and the initial configuration of the CA specify the evolution of CA that tells how each configuration is changed in one step.

2.2. Learning Automata

Learning Automata are adaptive decision-making devices that operate on unknown random environments. A learning Automaton has a finite set of actions to choose from and at each stage, its choice (action) depends upon its action probability vector. For each action chosen by the automaton, the environment gives a reinforcement signal with fixed unknown probability distribution. The automaton then updates its action probability vector depending upon the reinforcement signal at that stage, and evolves to some final desired behavior. A class of learning automata is called variable structure learning automata and are represented by quadruple $\{\alpha, \beta, p, T\}$ in which $\alpha = \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ represents the action set of the automata, $\beta = \{\beta_1, \beta_2, \dots, \beta_r\}$ represents the input set, $p = \{p_1, p_2, \dots, p_r\}$ represents the action probability set, and finally $p(n+1) = T[\alpha(n), \beta(n), p(n)]$ represents the learning algorithm. Let α_i be the action chosen at time n, then the recurrence equation for updating p is defined as

$$\begin{aligned} p_i(n+1) &= p_i(n) + a \cdot (1 - p_i(n)) \\ p_j(n+1) &= p_j(n) - a \cdot p_j(n) \quad \forall j \; j \neq i \end{aligned} \tag{1}$$

for favorable responses, and

$$\begin{aligned} p_i(n+1) &= (1-b) \cdot p_i(n) \\ p_j(n+1) &= \frac{b}{r-1} + (1-b) p_j(n) \quad \forall j \; j \neq i \end{aligned} \tag{2}$$

for unfavorable ones. In these equations, a and b are reward and penalty parameters respectively. If $a = b$, learning algorithm is called L_{R-P},[a] if $b \ll a$, it is called $L_{R\varepsilon P}$,[b] and if $b = 0$, it is called L_{R-I}.[c] For more information about learning automata the reader may refer to Refs. [57, 58].

2.3. Cellular Learning Automata

Cellular learning automata, which is a combination of cellular automata (CA) and learning automata (LA), is a powerful mathematical model for many decentralized problems and phenomena. The basic idea of CLA, which is a subclass of stochastic CA, is to utilize learning automata to adjust the state transition probability of stochastic CA. A CLA is a CA in which a learning automaton is assigned to every cell. The learning automaton residing in a particular cell determines its action (state) on the basis of its action probability vector. Like CA, there is a rule that the CLA operates under. The local rule of CLA and the actions selected by the neighboring LAs of any particular LA determine the reinforcement signal to the LA residing in a cell. The neighboring LAs of any particular LA constitute the local environment of that cell. The local environment of a cell is nonstationary because the action probability vectors of the neighboring LAs vary during evolution of the CLA. The basic idea of CLA, which is a subclass of stochastic CA, is to utilize learning automata to adjust the state transition probability of stochastic CA. A CLA is called synchronous if all LAs are activated at the same time in parallel. A CLA is called asynchronous (ACLA) if at a given time only some LAs are activated independently from each other, rather than all together in parallel. The LAs may be activated in either time-driven or step-driven manner. In time-driven ACLA, each cell is assumed to have an internal clock which wakes up the LA associated to that cell while in step-driven ACLA; a cell is selected in fixed or random sequence. CLA has found many applications such as image processing,[6–9] rumor diffusion,[10] modeling of commerce networks,[8] channel assignment in cellular networks[11] and VLSI placement,[12] to mention a few. For more information about CLA the reader may refer to Refs. [1, 10, 48, 49].

2.4. Irregular Cellular Learning Automata

An Irregular cellular learning automata (ICLA) (Fig. 1) is a cellular learning automata (CLA) in which the restriction of rectangular grid structure in traditional CLA is removed. This generalization is expected because there are applications such as wireless sensor networks, immune network systems, graph related applications, etc. that cannot be adequately modeled with rectangular grids. An ICLA is

[a]Linear Reward-Penalty.
[b]Linear Reward epsilon Penalty.
[c]Linear Reward Inaction.

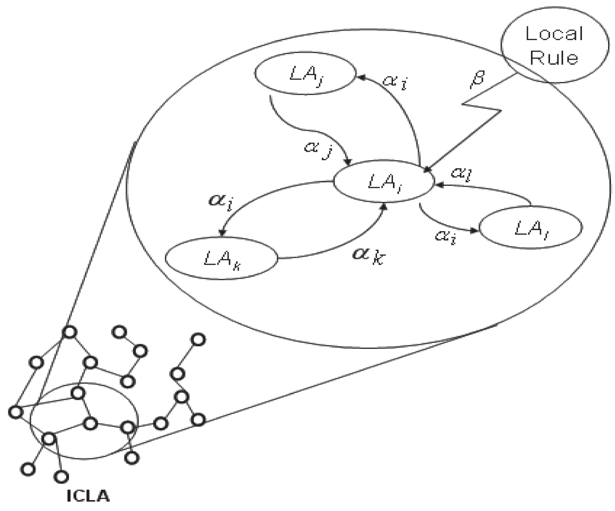


Fig. 1. Irregular cellular learning automata.

defined as an undirected graph in which, each vertex represents a cell which is equipped with a learning automaton. The learning automaton residing in a particular cell determines its state (action) on the basis of its action probability vector. Like CLA, there is a rule that the ICLA operate under. The rule of the CLA and the actions selected by the neighboring LAs of any particular LA determine the reinforcement signal to the LA residing in a cell. The neighboring LAs of any particular LA constitute the local environment of that cell. The local environment of a cell is non-stationary because the action probability vectors of the neighboring LAs vary during the evolution of the ICLA.

3. RELATED WORK

In this section, we will briefly overview some of the existing clustering algorithms for wireless sensor networks. The very first attempt to clustering sensor networks is the work of Nagpal and Coore[44] in 1998. The algorithm forms clusters with a maximum of two hops. Each node in the network takes part in the cluster formation process by choosing a random number from a fixed integer range. Then it counts down from that number silently. If the count down was not interrupted from any other neighboring node and it reaches zero, it announces itself as a head and broadcasts a *recruit* message. When a neighboring node receives the *recruit* message that comes within two-hop diameter boundary, it stops the count down, accepts the invitation and joins the cluster. A similar attempt is done by Bandyopadhyay and Coyle in Ref. [42]. In this method, each node becomes a head with probability p. Any node which resides within a maximum of k hops away from a head can join the cluster managed by that head. An analytical paradigm is given to find optimum p and k in terms of the whole energy consumption in the network.

Another pioneer attempt to make hierarchical infrastructures in wireless sensor networks is the one introduced[28] in 2001. Primarily, a tree rooted at one of the nodes of the network is created. Starting from leaves of this tree, each node counts number of its children and assigns them to different clusters based on a criterion which limits the number of nodes in each cluster to a pre-specified range. If number of children of a node doesn't reach the required amount for a cluster to be formed, it simply leaves clustering to its parent.

One of the most famous clustering algorithms introduced for sensor networks is the low energy adaptive clustering hierarchy (LEACH) algorithm.[15] In LEACH a predetermined fraction of nodes p, elect themselves as heads by comparing a chosen random number with a predefined threshold. After the heads have been elected, they broadcast an advertisement message to the rest of the nodes in the network that they are the new heads. Upon receiving this advertisement, all the non-cluster head nodes decide on the cluster to which they want to belong, based on the signal strength of the advertisement. In Ref. [34] an algorithm called CODA is presented which is based on the mechanism of LEACH. The difference is that clustering process is separately done in different distances (hop-based distance) or tiers to the sink node. Number of clusters in each tier is proportional to the distance of that tier to the sink node.

There are clustering algorithms which elect heads based on a certain criterion such as number of neighboring nodes,[16, 17, 37] or remaining energy of the nodes[18] or both.[45] In the algorithm presented in Refs. [19, 43], each node waits for a random duration. After this period, if no message is received from a certain head, the node states itself as a new head. In Ref. [26] an algorithm called Hybrid Energy-Efficient Approach (HEED) has been given. Before a node starts executing HEED, it sets its probability of becoming a head (CH_{prob}) based on its residual energy. Using this probability, each node decides to become a tentative head or not. Tentative heads, declare themselves to their neighbors using a cluster-head message, and double their CH_{prob} probability. Once CH_{prob} in a node reaches 1, that node becomes a final head. Nodes which are not head (tentative or final), upon receiving a cluster-head message, join to the declared cluster. In Ref. [27] an enhancement on HEED algorithm is presented.

In some clustering algorithms,[20, 21, 29–33] heads are known a priory and may differ from other nodes of the network. For such scenarios of pre-specified heads, Mhatre and Rosenberg in Refs. [35, 36] solve the optimality problem of number of heads and transmission mode in each cluster (single hop or multi-hop).

In Ref. [22], sink node is assumed to be a mobile node which queries data from different parts of the network. For this reason, it is required to have a dynamic clustering algorithm which can adapt itself to the changing position of the mobile sink, considering the overall energy consumption in the network. In a slightly different approach, Chen et al. in Ref. [38] propose a dynamic clustering method which is used to track a mobile target. Heads are known a priory and are always on, while other nodes are in sleep mode. When a number of heads detect the existence of a target, they first elect the nearest head to the target. Elected head requests its surrounding nodes to become active and sense the target. These nodes upon this request become active and send information of the position of the target to the elected head which then aggregates the received data and sends it to the sink node.

A number of different routing algorithms for sensor networks are surveyed in Refs. [23, 24]. One major class of these algorithms is the hierarchical routing. In hierarchical routing algorithms, at first a clustering scheme is applied to form a kind of hierarchy and afterward, the process of routing is divided into inter and intra-cluster phases. Some examples of these algorithms are "Fixed size cluster routing," TEEN,[d] and APTEEN.[e] AIMRP[f] Ref. [25] assumes that the sink node is placed in the center of the network. In the setup phase, sink initiates a Tire message to the nodes in the radius of a around itself. These nodes constitute the first level. These nodes rebroadcast the Tier message to the nodes in the radius of a around themselves. This process continues until all nodes in the network find their level in the hierarchy to the sink node.

Finally, two completely different clustering schemes proposed recently in Refs. [46 and 47]. Youssef et al. in Ref. [46] argue that although most of the published clustering algorithms strive to generate the minimum number of disjoint clusters, but guaranteeing some degree of overlap among clusters can facilitate many applications like inter-cluster routing, topology discovery and node localization and recovery from cluster head failure, etc. They proposed MOCA,[g] a randomized, distributed Multi-hop Overlapping Clustering Algorithm for organizing the sensors into overlapping clusters. The goal of the clustering process is to ensure that each node is either a head or within k hops from at least one head, where k is a preset cluster radius. Wang et al. in Ref. [47] promoted the idea of clustering the WSN based on the queries and attributes of the data. Clustering formation is started from sink node for every different attribute requested by the application. The result would be different clustering infrastructure for different attributes requested by the sink.

4. PROPOSED ALGORITHM

The proposed clustering algorithm consists of two phases: initial clustering and reclustering. In the initial clustering phase which is performed when the network starts operating, all nodes of the network participate. At the end of this phase a fully clustered network is created. Reclustering phase on the other hand, is initiated whenever the energy level of a cluster head node in the network degrades by a specified percent. In reclustering phase, only those nodes which are members of the cluster whose head initiates reclustering participate. The result of this phase is a new clustered infrastructure in that part of the network. Before the initial clustering phase starts, a time-driven asynchronous ICLA which is isomorphic to the sensor network topology is created. Each node s_i in the sensor network corresponds to the cell i in ICLA. Two cells i and j in ICLA are adjacent to each other if s_i and s_j in the sensor network are close enough to hear each other's signal. The learning automaton in each cell i of ICLA, referred to as LA_i, has two actions a_0 and a_1. Action a_1 is "declaring the node s_i as a head" and action a_0 is "declaring node s_i as a member." The probability of selecting each of these actions is initially set to 0.5. In the rest of this section algorithms for initial clustering phase and reclustering phase are described in details.

4.1. Initial Clustering Phase

During initial clustering the role of each node which is initially set to *unspecified* is changed to either *head* or *member*. Each node in the network maintains an array called *NeighborsInfo* including several pieces of information about each of its neighbors: energy level, number of its neighbors and the selected action by the corresponding cell. *NeighborsInfo* array for a node is empty at the beginning of the *initial clustering* phase and will be updated every time the node receives a packet called *ClusterADV* from any of its neighbors. *ClusterADV* packet will be introduced later in this section.

Each cell of ICLA during the *initial clustering* phase is activated asynchronously. The total number of times that a cell is activated is bounded by *MAX_ITERATION*. The nth activation of cell i occurs at time $R_i + n * t$ where R_i is a random number generated for cell i and t is a constant. We call n the local iteration number for the cell. Delays $\underline{R}_i$ are chosen randomly in order to reduce the probability of collisions between neighboring nodes.

The clustering algorithm starts when a cell in ICLA is activated. If cell i is activated then the following operations are performed:

—If the activation of cell i of ICLA is its first activation (local iteration 1) then the LA_i decides whether to declare node s_i in the sensor network as a head or as a member, that is, the cell chooses one of its actions using its action probability vector. Then the normal operation of the node s_i in the sensor network is interrupted and a packet called *ClusterADV* which contains action chosen by cell i, its energy level, and the number of entries in *NeighborsInfo* array (which is equal to the number of neighbors of node s_i from which *ClusterADV* packet is received so far) is created and transmitted to all of its neighbors

[d]Threshold-Sensitive Energy-Efficient Protocol.
[e]Adaptive Periodic TEEN.
[f]Address-light Integrated MAC, and Reporting Protocol.
[g]Multi-hop Overlapping Clustering Algorithm.

RESEARCH ARTICLE

—If the activation of cell i is not its first activation (local iteration $n > 1$) then

• The node s_i in the sensor network interrupts its normal operation and computes the reinforcement signal β_i according to Eq. (3) using the information maintained in its *NeighborsInfo* array.

$$\beta_i(n) = \begin{cases} 0; & -\alpha_i(n)\cdot\left(\sum_{j=1}^{N_i(n)}\alpha_j(n)\right)+(1-\alpha_i(n)) \\ & \cdot\left(\sum_{j=1}^{N_i(n)}\alpha_j(n)-1\right)\geq 0 \\ 1; & \text{Otherwise}\end{cases} \tag{3}$$

In Eq. (3), $N_i(n)$ is the number of neighbors of node s_i at local iteration n and $\alpha_i(n)$ is the selected action of LA_i at local iteration n. $\alpha_j(n)$ is the last action selected by cell j observed by node s_i (that is the action included in the last *ClusterADV* packet sent by node s_j and received by node s_i)

• LA_i updates its action probability vector using the computed β_i according to the following learning algorithm with time varying parameters $a(n)$ and $b(n)$.

$$\begin{aligned} p_i(n+1) &= p_i(n)+a(n)\cdot(1-p_i(n)) \\ p_j(n+1) &= 1-p_i(n+1)\end{aligned} \tag{4}$$

$$\begin{aligned} p_i(n+1) &= (1-b(n))\cdot p_i(n) \\ p_j(n+1) &= 1-p_i(n+1)\end{aligned} \tag{5}$$

Reward parameter $a(n)$ and penalty parameter $b(n)$ are time varying parameters which vary according to Eqs. (6) and (7). Values of these two parameters at time n at a given cell depend on the current selected action by the cell, current energy level and the number of the neighbors of the corresponding node.

$$a(n) = \alpha_i(n)\cdot(\phi\cdot\psi_i(n))+(1-\alpha_i(n))\cdot[\phi\cdot(1-\psi_i(n))] \tag{6}$$

$$b(n) = \alpha_i(n)\cdot[\rho\cdot(1-\psi_i(n))]+(1-\alpha_i(n))\cdot(\rho\cdot\psi_i(n)) \tag{7}$$

where

$$\psi_i(n) = \frac{E_{s_i}(n)}{MaxEnergyLevel} \tag{8}$$

In Eqs. (6) and (7) $\alpha_i(n)$ is the action selected by the cell at time n and φ and ρ are two constants. It can be shown that If the reward parameter $a(n)$ and the penalty parameter $b(n)$ vary according to Eqs. (6) and (7) and ϕ, $\rho < 1$, then we have $0 < a(n) < 1$, $0 < b(n) < 1$ (Lemma 1 in appendix). In Eq. (8), $E_{s_i}(n)$ is the residual energy level of the node s_i at time n and *MaxEnergyLevel* is the energy level of a full battery charged node.

• Cell i chooses one of its actions using its action probability vector; that is declare the node s_i in the sensor network as a head or as a member.

• Node s_i creates a *ClusterADV* packet and transmits it to its neighbors.

—Node s_i resumes its normal operation.

Initial clustering phase for a node stops if either its local iteration number exceeds *MAX_ITERATION* or if the role of the node changes from *unspecified* to head or member. A node becomes a head if the probability that the node declares itself as a head exceeds a predetermined threshold (*UP_THRESHOLD*) and becomes a member if it receives a signal (contained in *ClusterADV* packet) from a node which has already become a head.

If the clustering phase for a node is over because it's local iteration number exceeds *MAX_ITERATION* but its role is still *unspecified*, then it polls all of its neighbors whether or not any of them is a head. If there are such neighbors, then the polling node becomes a member of the cluster of one of those neighbors chosen at random. Otherwise, the polling node itself becomes a head.

4.2. Reclustring Phase

Reclustering phase in many clustering algorithms reported in the literatures is performed periodically for the entire network.[15,27,45] In such algorithms, in predetermined time intervals, the normal operation of the network is interrupted, the clustering algorithm is performed on the entire network (producing a completely new clustered infrastructure) and then the normal operation of the network is resumed. Such periodical reclustering schemes has a number of drawbacks. The very first problem with such schemes is that they consume too much energy because the reclustering is performed on the entire network. Another problem is that the normal operation of the network is delayed until the reclustering phase is over. In many critical applications such as military domains, this delay is not acceptable. Finally, such reclustering schemes are static and do not receive any feedback from the network and the state of its nodes. Periodical reclustering may change the role of a node from a head to a member though it still has enough residual energy to continue its role as a head. Unlike such static reclustering schemes, the reclustering scheme proposed in this paper is performed locally and adaptively whenever it is needed. A local reclustering is only required if a head consumes so much energy that cannot continue its role as a head. In such a situation, a local reclustering is initiated in the cluster managed by that head. The role of the head is transferred to the newly elected head(s) at the end of the local reclustering phase, thus normal operation of the network can be intermingled with the reclustering phase.

Reclustering phase is initiated by a head whenever its energy level degrades to a specified percent (*RECLUSTERING_PERCENT*) below the energy level at the time it was selected as a head. Upon the initiation of a reclustering phase by a node s_i whose role is a head, the following operations are performed:

—s_i creates a packet called *Recluster* and transmits it to its neighbors. This packet is just for informing the neighbors of s_i the beginning of reclustering phase and contains no other information.

—If the neighbor s_j received the *Recluster* packet is a head then neighbor s_j creates a *ClusterADV* packet and transmits it to its neighbors.

—If neighbor s_k received the *Recluster* packet is a member then

- If neighbor s_k is a member of the cluster managed by s_i then
- The action probability vector (p_1, p_2) of LA_k is set to (0.5, 0.5)
- Neighbor s_k changes its role from a member to *unspecified*
- Neighbor s_k starts to execute the initial clustering phase algorithm with smaller value for parameter *MAX_ITERATION*.
- If neighbor s_k is not one of the members of the cluster managed by s_i then
- Neighbor s_k disregards the received *Recluster* packet and continues its normal operation.

Once the reclustering phase is over, members of the cluster whose energy was depleted may have become a head or a member of an existing cluster or a member of a newly formed cluster. During the reclustering phase, previous head (node s_i) maintains its role as a head to allow network performs its normal operation in parallel to reclustering.

Some drawbacks and limitations of the proposed method which may be considered for further studies are as follows:

—As the density of the network increases, the overhead of receiving *ClusterADV* messages from neighbors in each node increases.

—Although the proposed clustering algorithm can be utilized in tracking or query based applications, but it seems a more dynamic clustering method can better fit such applications.

—The proposed clustering algorithm assumes that all of the sensor nodes have the ability to change their transmission ranges from R_c to $3R_c$ which is not always the case.

—The proposed clustering algorithm assumes that the links in the network are bidirectional but this assumption due to hidden terminal problem[59] or due to disparity between the transmission power levels of the nodes at the end of the links, may not hold.

5. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed clustering algorithm, a number of experiments have been conducted. In these experiments, the results obtained from the proposed algorithm are compared with the results obtained from the LEACH,[15] the basic HEED[26] and its extension (ExtendedHEED).[27] In these experiments, we study the quality of the proposed clustering algorithm in terms of *Number of clusters* in the clustering infrastructure produced at the end of the clustering, *Percentage of the sparse clusters* which is defined as the percentage of clusters having only one head and one member, *Mean energy level of heads* and *Ratio of the mean energy level of heads to the mean energy level of members* defined by Eq. (9) where n_h is the number of heads and n_m is the number of members.

$$\zeta = \frac{\sum_{i=1}^{n_\mathrm{h}} E_{s_i}}{\sum_{j=1}^{n_\mathrm{m}} E_{s_j}} \tag{9}$$

In what follows, we first give the simulation scenario used in our experiments and then give the simulation results.

5.1. Simulation Scenario

In the simulation scenario used in our experiments, the clustering algorithm is performed first to determine the role of each node of the network as a head or as a member. In the clustering algorithm, nodes communicate with each other using R_c as their transmission range. Then, the normal operation of the network which is assumed to be a simple monitoring application is started. A simple monitoring application is defined to be an application in which members of each cluster periodically obtain data from the environment and send it to the head of the cluster. The head of each cluster periodically aggregate data received from its members and then send the aggregated data towards the sink using a multi-hop inter-cluster routing scheme. Inter-cluster communications are performed using transmission range of $3R_c$. In such an application, a data gathering round is defined to be the time during which the sink node receives one single packet from each head of the network. It can be shown that in such applications, the optimum number of clusters in order to minimize the total energy consumption of the network at each data gathering round is $k = d^{(1-d)\cdot((E_s^f - E_s^n - E_R)/(E_s^f + E_R))+1/(d-1)-1/\ln(d)+2}$ where d is the averaged branching factor of the routing tree used for routing, E_S^f and E_S^n are the amount of energy consumed for sending a data packet using transmission range of $3R_c$ and R_c respectively and E_R is the amount of energy consumed for receiving a data packet (Theorem 1 in the appendix).

For the normal operation of the network to be performed correctly, heads must be able to communicate with each other. It can be shown that using the transmission range of $3R_c$ for inter-cluster communications, a connected graph of all heads in the network is produced by the proposed clustering algorithm asymptotically almost surely (Theorem 2 in the appendix).

The network operation continues until the time at which a node in the network dies. In other words, network lifetime is defined to be the time elapsed from the network startup to the time at which a node in the network dies.[19]

5.2. Simulation Results

In the simple monitoring application considered in our experiments, nodes periodically report 525 bytes of data to the sink node. Directed diffusion[41] is used as the multi-hop inter-cluster routing protocol. All simulations have been implemented using NS2 simulator. We use IEEE 802.11 as the MAC layer protocol. Nodes are placed randomly on a 2 dimensional area of size 100(m) × 100(m). In all experiments, initial random delay R_i is selected uniformly and randomly in the range [0.06, 0.09] (s), t is set to 10 (s), *MAX_ITERATION* is set to 50, φ is set to .8, ρ is set to 0.2, R_c is set to 20(m), *MaxEnergyLevel* is set to 2.0(J) and *RECLUSTERING_PERCENT* is set to 85%. *MAX_ITERATION* for reclustering phase is set to 10 and initial energy level of nodes is selected uniformly and randomly in range [1.5, 2.0] (J). First order radio model specified in Ref. [15] is used for estimating the amount of energy consumed for packet transmissions. All packets except for data packets, which are 525 bytes long, are assumed to be 8 bytes. For the sake of simplicity we assume that energy consumption in idle states is zero. Simulations are performed for 50, 100, 200, 300, 400, and 500 nodes. The results are averaged over 50 runs.

Experiment 1. In this experiment, we compare the quality of the clustering infrastructure produced using the proposed method, LEACH, HEED and ExtendedHEED in terms of number of clusters, percentage of sparse clusters, mean energy level of heads and ratio of the mean energy level of heads to the mean energy level of members (ζ). Figure 2 compares the number of generated clusters using the proposed method and the existing methods with the optimum number of clusters (Theorem 1). As it is shown the number of clusters in the infrastructure resulted from the proposed clustering algorithm is very close to the optimum number of clusters. Figures 3, 4 and 5 compares the proposed algorithm with the other algorithms in terms of percentage of sparse clusters, mean energy level of heads and ζ, respectively. These figures show that the clustering infrastructure formed by the proposed method is better than the other three methods in terms of all of these parameters.

Experiment 2. This experiment whose result is given in Figure 6 compares the network lifetime for the proposed algorithm with LEACH, HEED and ExtendedHEED algorithms. Figure 6 shows that the network lifetime

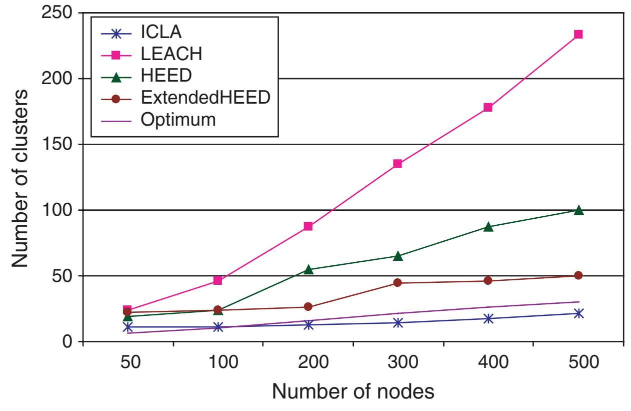


Fig. 2. Comparison of the number of clusters resulted from the clustering methods with the optimum number of clusters given by Theorem 1.

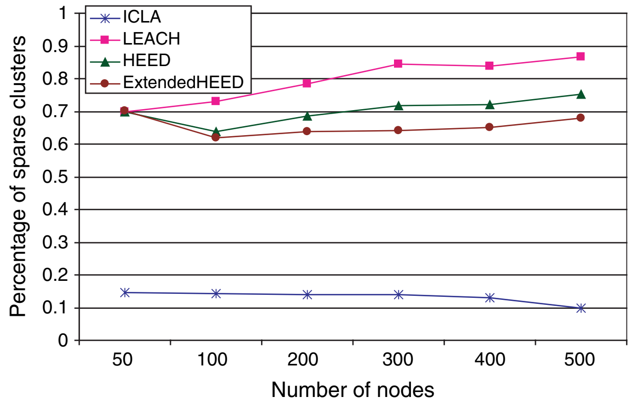


Fig. 3. Comparison of clustering methods with respect to *Percentage of sparse clusters*.

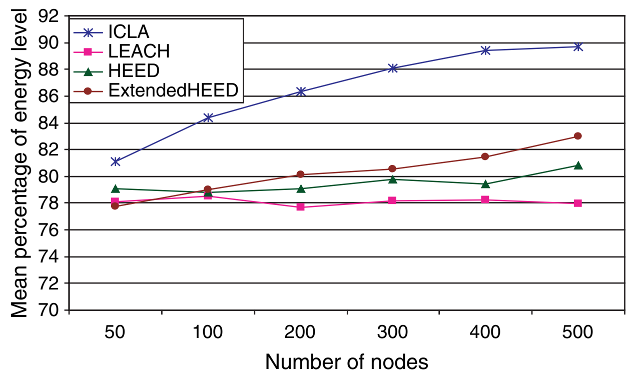


Fig. 4. Comparison of clustering methods with respect to *Mean energy level of heads*.

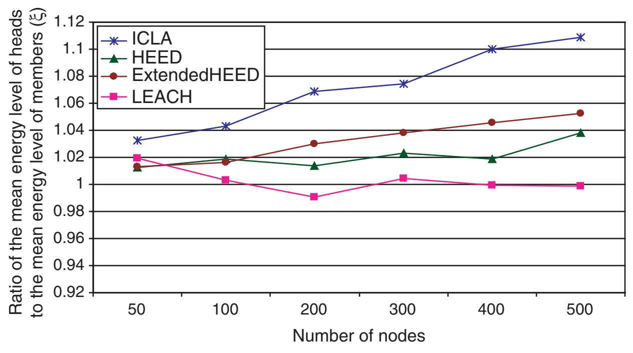


Fig. 5. Comparison of clustering methods with respect to Ratio of the mean energy level of heads to the mean energy level of members (ζ).

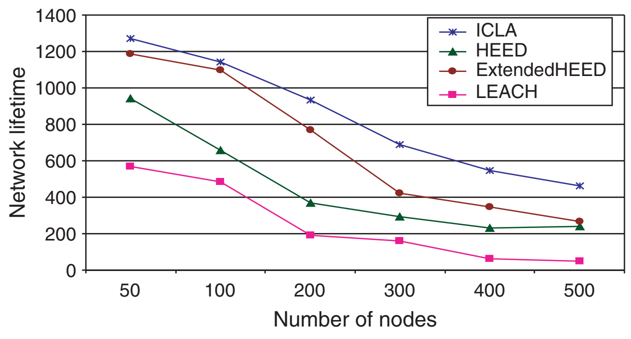


Fig. 6. Comparison of clustering methods with respect to network lifetime.

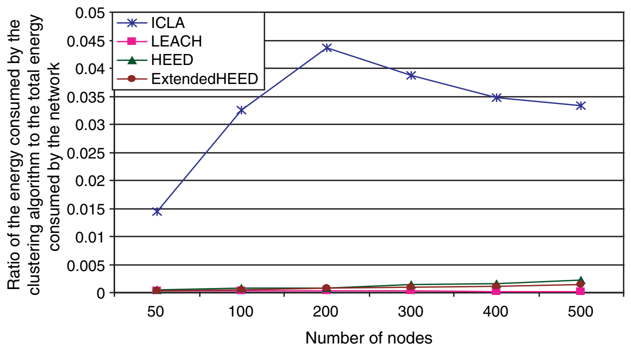


Fig. 7. Comparison of clustering methods with respect to the Ratio of the energy consumed by the clustering algorithm to the total energy consumed by the network

for the proposed algorithm is higher than that for other algorithms.

Experiment 3. In this experiment we study the proposed clustering algorithm in terms of energy consumption. For this purpose, we compare the ratio of the energy consumed by the clustering algorithm to the total energy consumed by the network for the proposed clustering algorithm and existing algorithms. Figure 7 gives the result of this comparison. As it is shown, the proposed clustering algorithm performs worse than the existing methods in terms of the energy consumption. But careful inspection of the results given by Figure 7, shows that in the worst case the amount of energy consumed during the clustering phase in comparison to the total energy dissipated during the network lifetime is negligible. The amount of energy consumed during the clustering phase is about 0.0004 of the total energy dissipated during the network lifetime.

6. CONCLUSIONS

In this paper, we first propose a generalization of cellular learning automata (CLA) called irregular cellular learning automata (ICLA) and then a new clustering algorithm for sensor networks based on ICLA was proposed. To evaluate the performance of the proposed method, several experiments were conducted using the proposed clustering algorithm and the results obtained were compared with the results obtained for LEACH, the basic HEED and its extension (ExtendedHEED) in terms of *Number of clusters* in the clustering infrastructure produced at the end of the clustering, *percentage of the sparse clusters*, *mean energy level of heads* and *ratio of the mean energy level of heads to the mean energy level of members*. Experiments showed that the proposed clustering algorithm outperforms the existing clustering algorithms.

APPENDIX

Lemma 1. If the reward parameter $a(n)$ and the penalty parameter $b(n)$ vary according to Eqs. (6) and (7) and ϕ, $\rho < 1$, then we have $0 < a(n) < 1$ and $0 < b(n) < 1$.

Proof. Using Eq. (8) and knowing the fact that $E_{s_i}(n) \leq$ *MaxEnergyLevel* for every node s_i, then we have $\psi_i(n) \leq 1$. Using Eqs. (6) and (7) and knowing the fact that $\alpha_i(n)$ is either 0 or 1, $\psi_i(n) \leq 1$, and ϕ, $\rho < 1$ we can conclude that $a(n) < 1$ and $b(n) < 1$.

Lemma 2. Assume that the normal operation of the network is a simple monitoring application which is performed using a multi-hop inter-cluster routing scheme. Also assume that the routing scheme uses a routing tree rooted at the sink node, spanned throughout the network and has a branching factor of d on average. Then on average each head in this routing tree relays $\eta = (d^2 - (1 + (1-d)\cdot(\log_d^k - 2))\cdot k)/(k\cdot(1-d)^2)$ packets of other heads where k is the number of clusters in the network.

Proof. Consider the routing tree depicted in Figure 8.

The root of the tree is the sink and other nodes are head nodes. Any level of this tree relays the packets received from the levels beneath it. Assuming that each node has an average branching factor of d then the average number of packets relayed at level i of this tree (μ_i) and the total average number of packets relayed in the network ($ATPR$) are given by Eqs. (10) and (11), respectively.

$$\frac{d^{i+1} - d^{L+1}}{1-d} = d^{i+1} + d^{i+2} + \cdots + d^L = \mu_i \tag{10}$$

$$ATPR = \sum_{i=1}^{L} \mu_i = \frac{d^2 - (1 + (1-d)\cdot(L-1))\cdot d^{L+1}}{(1-d)^2} \tag{11}$$

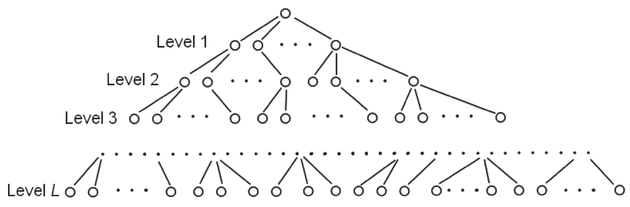


Fig. 8. Routing tree used for inter-cluster routing.

Using Eq. (11) the average number of packets relayed by each node in the routing tree is then computed using Eq. (12) given below

$$\eta = \frac{ATPR}{k} = \frac{d^2 - (1+(1-d)\cdot(L-1))\cdot d^{L+1}}{k\cdot(1-d)^2} \tag{12}$$

where k is the number of clusters in the network. Replacing L by $\log_d^k - 1$ in Eq. (12) we get

$$\eta = \frac{d^2 - (1+(1-d)\cdot(\log_d^k - 2))\cdot k}{k\cdot(1-d)^2} \tag{13}$$

Theorem 1. Assume that the normal operation of a network to be a simple monitoring application and the energy consumption for reclustering phase is zero. Then if the transmission range for intra-cluster communication and inter-cluster communication are R_c and $3R_c$, respectively then the optimum number of clusters (k) in order to minimize the total energy consumed by the network at each round of data gathering is $k = d^{(1-d)\cdot((E_s^f - E_s^n - E_R)/(E_s^f + E_R)) + 1/(d-1) - 1/\ln(d) + 2}$, where d is the average branching factor of the routing tree, E_S^f and E_S^n are the amount of energy consumed for sending a data packet using transmission range of $3R_c$ and R_c respectively and E_R is the amount of energy consumed for receiving a data packet.

Proof. Let E_{ch} be the amount of energy consumed by a head and E_{cm} be the amount of energy consumed by a member at each round of data gathering. Since the amount of energy consumed in reclustering phase is assumed to be zero, E_{ch} can be expressed using Eq. (14).

$$E_{\text{ch}} = \left(\frac{N}{k} - 1\right)\cdot E_R + E_S^f + \eta\cdot(E_R + E_S^f) \tag{14}$$

E_{ch} consists of three terms; $((N/k) - 1)\cdot E_R$ is the amount of energy consumed for receiving data from the members of the head (number of members in a cluster is $(N/k) - 1$ on average), E_S^f is the amount of energy consumed for sending aggregated data towards the sink, and $\eta\cdot(E_R + E_S^f)$ is the amount of energy consumed for relaying messages received from other heads. η is the average number of packets relayed by each head (Lemma 2). The amount of energy consumed for aggregating data packets received from members is neglected.

A member only sends its reading to the head of its cluster using R_c as its transmission range and hence, $E_{\text{ch}} = E_S^n$. Therefore the total energy dissipated in the network at each round of data gathering ($E(k)$) can be given by Eq. (15) where k is the number of clusters.

$$\begin{aligned} E(k) &= k\cdot E_{\text{ch}} + (N-k)\cdot E_{\text{cm}} \\ &= k\cdot\left[\left(\frac{N}{k} - 1\right)\cdot E_R + E_S^f + \eta\cdot E_R + \eta\cdot E_S^f\right] \\ &\quad + (N-k)\cdot[E_S^n] \end{aligned} \tag{15}$$

To find the number of clusters which minimize $E(k)$ we set the derivative of $E(k)$ with respect to k to zero, that is

$$\frac{dE(k)}{dk} = 0 \tag{16}$$

Since the transmission range for intra-cluster communication and inter-cluster communication are R_c and $3R_c$ respectively then E_R, E_S^f and E_S^n are constants and hence we have

$$\frac{dE(k)}{dk} = E_S^f - E_S^n - E_R + (E_R + E_s^f)\cdot\frac{d(k\cdot\eta)}{dk} = 0 \tag{17}$$

From Eq. (13), we have

$$\frac{d(k\cdot\eta)}{dk} = \frac{d}{dk}\left(\frac{d^2 - (1+(1-d)\cdot(\log_d^k - 2))\cdot k}{(1-d)^2}\right) \tag{18}$$

or

$$\frac{d(k\cdot\eta)}{dk} = \frac{1}{d-1}\cdot\left(\frac{1}{1-d} + \frac{1}{\ln(d)} + (\log_d^k - 2)\right) \tag{19}$$

After replacing $d(k\cdot\eta)/dk$ from Eq. (19) in Eq. (17) we have

$$\begin{aligned} \frac{dE(k)}{dk} &= E_S^f - E_S^n - E_R + \frac{(E_R + E_s^f)}{d-1} \\ &\quad \cdot\left(\frac{1}{1-d} + \frac{1}{\ln(d)} + (\log_d^k - 2)\right) = 0 \end{aligned} \tag{20}$$

Solving the above equation for k we obtain

$$k = d^{(1-d)\cdot((E_s^f - E_s^n - E_R)/(E_s^f + E_R)) + 1/(d-1) - 1/\ln(d) + 2} \tag{21}$$

Lemma 3. At the termination of the initial clustering phase of the proposed algorithm, a node is either tagged as a head or as a member.

Proof. Initial clustering in a node stops either when the number of iterations reaches *MAX_ITERATION* or when the role of the node changes from *unspecified* to a head or a member. In the former case, the role of the node is still unknown. Such a node then polls all of its neighbors whether or not any of them is a head. If there are such neighbors, then the polling node becomes a member of the cluster of one of those neighbors chosen at random. Otherwise, the polling node itself becomes a head. So, at the end of the initial clustering phase of the proposed algorithm, a node is either tagged as a head or as a member.

Lemma 4. Assume that N nodes are uniformly and independently dispersed at random in an area $R = [0, L]^d$ for $d = 1, 2, 3$ and assume that $R_c^d N = aL^d \ln L$ for some constant $a > 0$, with $R_c \ll L$ and $N \gg 1$. If $a > d\cdot a_d$, or $a = d\cdot a_d$ and $R_c \gg 1$, then $\lim_{L\to\infty} P_{\text{conn}}(L) = 1$, where $a_d = 2^d d^{d/2}$ and $P_{\text{conn}}(L)$ denotes the probability that the communication graph is connected.

Proof. The proof of the above lemma is given in Ref. [40].

Corollary 1. If the conditions of Lemma 4 hold for a 2 dimensional network area, then each cell of size $R_c/\sqrt{2} \times R_c/\sqrt{2}$ contains at least one node a.a.s.

Proof. The proof is trivial from Lemma 4.

Theorem 2. Assume that the conditions of *Lemma 4* hold. Then, if transmission range of $3R_c$ is used for inter-cluster communications, a connected graph of all heads in the network is produced by the proposed algorithm asymptotically almost surely.

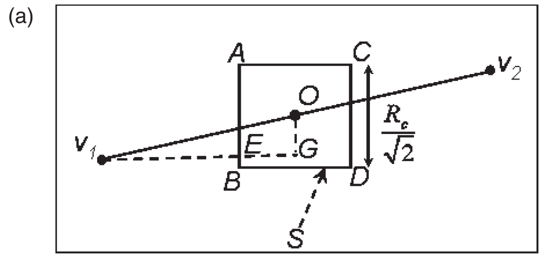
Proof. Assume that the proposed clustering algorithm produces two connected sub-graphs of heads $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, such that any $v_1 \in V_1$ cannot communicate with any $v_2 \in V_2$ using transmission range of $3R_c$. We prove the theorem by showing that such a situation cannot happen. Assume that v_1 and v_2 are the two nearest heads in G_1 and G_2. Since v_1 and v_2 cannot communicate with each other, their distance is more than $3R_c$. Now consider a square S of size $(R_c/\sqrt{2}) \times (R_c/\sqrt{2})$ centered on the middle of the line passes through v_1 and v_2 and its edges are parallel to the edges of the area R (Fig. 9). It is clear that the nearest point on square S to v_1 is on the nearest edge of S to v_1 (edge AB in Fig. 9). Two cases can be considered:

Case 1: when the perpendicular line from v_1 to AB crosses AB in a point within AB (point E in Fig. 9(a)), then E is the nearest point to v_1 and therefore $d(v_1, E)$, the distance between points v_1 and E, is given by Eq. (28)

$$d(v_1, E) = \sqrt{d^2(v_1, O) - d^2(O, G)} - d(E, G) \qquad (28)$$

Using Eq. (28) and the fact that$d(v_1, v_2) \geq 3R_c$ we get

$$d(v_1, E) = \sqrt{\frac{9R_c^2}{4} - d^2(O, G)} - \frac{R_c}{2\sqrt{2}} \qquad (29)$$



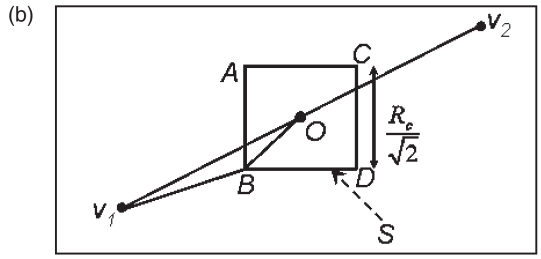


Fig. 9. v_1 and v_2 are two heads in two disconnected graphs of heads and square S is centered on O which is in the middle of the line v_1v_2. (a) The nearest point on square S to v_1 is E, (b) The nearest point on square S to v_1 is B.

To obtain the minimum of $d(v_1, E)$, we let $d(O, G)$ to have its maximum value which is $R_c/(2\sqrt{2})$. Therefore we get

$$d(v_1, E) \geq \frac{\sqrt{17}-1}{2\sqrt{2}} \cdot R_c \geq R_c \qquad (30)$$

Case 2: when the perpendicular line from v_1 to AB doesn't cross AB in a point within AB (Fig. 9(b)), then the nearest point to v_1 is either A or B (B in Fig. 9) and therefore $d(v_1, B)$, the distance between points v_1 and B, can be calculated using law of cosines as follows:

$$d(v_1, B) = \sqrt{d^2(v_1, O) + d^2(O, B) - 2 \cdot d(v_1, O) \cdot d(O, B) \cdot \cos(\widehat{BOv_1})} \qquad (31)$$

To obtain the minimum of $d(v_1, B)$ we set angle $\widehat{BOv_1}$ to 0°. Therefore we get

$$d(v_1, B) \geq R_c \qquad (32)$$

Since in the proposed clustering algorithm, members of a cluster can communicate with the head of the cluster using transmission range of R_c (i.e., members of a cluster are within the distance of at most R_c from the head of the cluster), the above two cases immediately imply that no point on square S can be a member of v_1. Using the same argument, it can be shown that no point on S can be a member of v_2 as well. This means that any sensor node within the square S cannot be a member of either v_1 or v_2.

On the other hand, assuming that the conditions of Lemma 4 hold, using Corollary 1 we conclude that at least one node say node s_i exists in square S a.a.s. As s_i located within square S, it cannot be a member of either v_1 or v_2. Thus according to Lemma 3, s_i is either a head or a member of another cluster managed by a head say v_3 (Fig. 10). We next show that s_i cannot be a head or a member of a cluster managed by v_3.

If s_i is a head, then either $s_i \in V_1$ or $s_i \in V_2$. Without loss of generality, assume $s_i \in V_1$. Since, s_i located on square S, it is clear that $d(s_i, v_2) < d(v_1, v_2)$. This contradicts with the assumption that v_1 and v_2 are the two nearest heads in G_1 and G_2 and hence s_i cannot be a head.

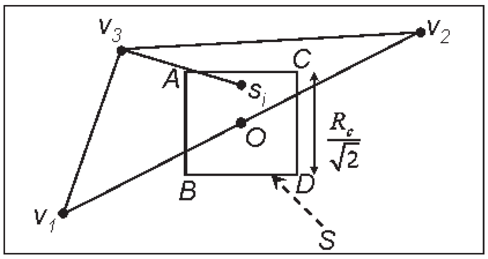


Fig. 10. A sensor node s_i in square S must be either a head or a member of a cluster managed by a head say v_3.

If s_i is a member of a cluster managed by v_3, then either $v_3 \in V_1$ or $v_3 \in V_2$. Without loss of generality, assume $v_3 \in V_1$. We can show that $d(v_3, v_2) < d(v_1, v_2)$ which contradicts with the assumption that v_1 and v_2 are the two nearest heads in G_1 and G_2. $d(v_3, v_2)$ can be calculated using the following equation:

$$d(v_3, v_2) = \sqrt{d^2(v_3, O) + d^2(O, v_2) - 2 \cdot d(v_3, O) \cdot d(O, v_2) \cdot \cos(\widehat{v_3 O v_2})} \quad (33)$$

Having the fact that $d(v_3, O) < (3/2)R_c$ and considering $d(v_1, v_2) = 3R_c + \varepsilon$ for some $\varepsilon > 0$, $d(v_3, v_2)$ can be computed as

$$d(v_3, v_2) \le \sqrt{\frac{9}{4}R_c^2 + \frac{(3R_c + \varepsilon)^2}{4} - 3R_c\left(\frac{3R_c + \varepsilon}{2}\right) \cdot \cos(\widehat{v_3 O v_2})} \quad (34)$$

To obtain the maximum of $d(v_3, v_2)$ we set the angle $\widehat{v_3 O v_2}$ to 180°. Therefore we get

$$d(v_3, v_2) \le 3R_c + \frac{\varepsilon}{2} \quad (35)$$

Inequality (35) shows that $d(v_3, v_2) < d(v_1, v_2)$. Thus, no head like v_3 can be found which manages a cluster to which the node s_i is assigned.

If s_i is not a head or a member, then the status of s_i is *unspecified* which is in contradiction with the result of Lemma 3. Therefore G_1 and G_2 are connected a.a.s. and hence the theorem.

References and Notes

1. H. Beigy and M. R. Meybodi, *Advances in Complex Systems* 7, 295 **(2004)**.
2. E. Fredkin, *Physica D* 45, 245 **(1990)**.
3. M. Mitchell, Computation in cellular automata: A selected review, Technical report, Santa Fe Institute, Santa Fe, NM, USA, September **(1996)**.
4. N. H. Packard and S. Wolfram, *J. Stat. Phys.* 38, 901 **(1985)**.
5. J. Kari, *Physica D* 45, 379 **(1990)**.
6. M. R. Kharazmi and M. R. Meybodi, *Proc. of 10th Iranian Conf. on Electrical Engineering, ICEE-96*, Tabriz, Iran, May **(2001)**.
7. M. R. Kharazmi and M. R. Meybodi, *Proc. of 2nd Iranian Conf. on Machine Vision, Image Processing and Applications*, Tehran, Iran **(2003)**, pp. 261–270.
8. M. R. Meybodi, H. Beygi, and M. Taherkhani, *J. Sci. Tech.* 54 **(2004)**.
9. M. R. Meybodi and M. R. Khojaste, *Proc. of 6th Annual Intl. Computer Society of Iran Computer Conf., CSICC-2001*, Isfehan, Iran, February **(2001)**, pp. 284–295.
10. M. R. Meybodi and M. Taherkhani, *Proc. of 9th Conf. on Electrical Engineering*, Power and Water Institute of Technology, Tehran, Iran, May **(2001)**, pp. 102–110.
11. H. Beigy and M. R. Meybodi, Lecture Notes in Computer Science, Springer-Verlag **(2003)**, Vol. 2690, pp. 119–126.
12. H. Beigy and M. R. Meybodi, Lecture Notes in Computer Science, Springer-Verlag **(2002)**, Vol. 2510, pp. 450–457.
13. H. Couclelis, *Environment and Planning* 585 **(1985)**.
14. D. Stevens, Integration of an irregular cellular automata approach and geographic information systems for high-resolution modeling of urban growth. M.Sc. thesis, Department of Geography, University of Toronto **(2003)**.
15. W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, *Intl. Conf. on System Sciences*, Hawaii, January **(2000)**.
16. C. Y. Wen and W. A. Sethares, *EURASIP Journal on Wireless Communications and Networking* 686 **(2005)**.
17. K. Shin, A. Abraham, and S. Y. Han, *Intl. Conf. on Computational Science and Applications, Lecture Notes in Computer Science (LNCS 3983)*, edited by M. Gavrilova et al., Springer Verlag, Germany, UK **(2006)**, pp. 40–49.
18. M. Ye, Ch. Li, G. Chen, and J. Wu, *Proc. of the IEEE Intl. Workshop on Strategies for Energy Efficiency in Ad Hoc and Sensor Networks (IWSEEASN'05)*, April **(2004)**.
19. V. Mittal, M. Demirbas, and A. Arora, LOCI: Local clustering service for large scale wireless sensor networks, Technical Report OSU-CISRC-2/03-TR07, Ohio State University **(2003)**.
20. S. Soro and W. B. Heinzelman, *5th Intl. IEEE Workshop on Algorithms for Wireless, Mobile, Ad Hoc, and Sensor Networks (WMAN05)*, April **(2005)**.
21. Y. Guo and J. MeNair, *Proc. of the Intl. Conf. on Cybernetics and Information Technologies, Systems, and Applications (CITSA)*, Orlando, Florida, July **(2004)**.
22. M. Lotfinezhad and B. Liang, *Proc. of the IEEE Wireless Communications and Networking Conference (WCNC)*, New Orleans, Louisiana, March **(2005)**.
23. K. Akkaya and M. Younis, *Elsevier Ad Hoc Network Journal* 325 **(2005)**.
24. M. Ilyas and I. Mahgoub, Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems, CRC Press, London, Washington, DC **(2005)**.
25. S. Kulkarni, A. Iyer, and C. Rosenberg, *IEEE/ACM Transactions on Networking*, accepted August **(2005)**.
26. O. Younis and S. Fahmy, *Proc. of IEEE INFOCOM*, March **(2004)**, Vol. 1, pp. 629–640.
27. H. Huang and J. Wu, *Proc. of IEEE 62nd Semiannual Vehicular Technology Conference (VTC)*, September **(2005)**.
28. S. Banerjee and S. Khuller, *Proc. Of the 20th Annual Joint Conf. of the IEEE Computer and Communications Societies (INFOCOM)*, Anchorage, AK, USA **(2001)**, pp. 1028–1037.
29. A. Ghiasi, A. Srivastava, X. Yang, and M. Sarrafzadeh, *Sensors Magazine, MDPI*, January **(2002)**, Issue 1, pp. 258-269.
30. G. Gupta and M. Younis, *Proc. of the Intl. Conf. on Communication (ICC 2003)*, Anchorage, Alaska, May **(2003)**.
31. G. Gupta and M. Younis, *Proc. of the 10th IEEE Int'l Conf. on Telecommunications (ICT)* **(2003)**, pp. 1577–1583.
32. G. Gupta and M. Younis, *In Proc. of the IEEE Wireless Communication and Networks Conf. (WCNC 2003)*, New Orleans, Louisiana, March **(2003)**.
33. M. Younis, M. Youssef, and Kh. Arisha, *Computer Networks*, Elsevier **(2003)**, pp. 649–668.
34. S. H. Lee, J. J. Yoo, and T. Ch. Chung, *Proc. Of the 29th Annual IEEE Intl. Conf. on Local Computer Networks*, November **(2004)**, pp. 567–568.
35. V. Mhatre and C. Rosenberg, *Ad Hoc Networks*, Elsevier **(2004)**, pp. 45–63.
36. V. Mhatre and C. Rosenberg, *Proc. Of IEEE Intl. Conf. on Communications*, June **(2004)**, pp. 3646–3651.
37. H. Chan and A. Perrig, *Proc. of First European Workshop Sensor Networks (EWSN)*, January **(2004)**.
38. W. P. Chen, J. C. Hou, and L. Sha, *Proc. of IEEE Intl. Conf. on Network Protocols*, November **(2003)**.

39. B. Liu and D. Towsley, *Proc. of the Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks Conference (WiOpt)* (**2003**).

40. D. M. Blough and P. Santi, *Proc. of the ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MOBICOM)* (**2002**).

41. C. Intanagonwiwat, R. Govindan, and D. Estrin, *IEEE/ACM Trans. Networking* 2 (**2003**).

42. S. Bandyopadhyay and E. Coyle, *Proc. of the 22nd Annual Joint Conf. of the IEEE Computer and Communications Societies (INFOCOM 2003)*, San Francisco, California, April (**2003**).

43. M. Demirbas, A. Arora, and V. Mittal, *Proc. of Workshop on Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks (DIWANS'04)*, Palazzo dei Congressi, Florence, Italy, June (**2004**).

44. R. Nagpal and D. Coore, *Proc. of the 10th Intl. Conf. on Parallel and Distributed Systems (PDCS'98)*, Las Vegas, NV, October (**1998**).

45. P. Ding, J. Holliday, and A. Celik, *Proc. of the IEEE Intl. Conf. on Distributed Computing in Sensor Systems(DCOSS'05)*, Marina Del Rey, CA, June (**2005**).

46. A. Youssef, M. Younis, M. Youssef, and A. Agrawala, *Proc. of the 49th Annual IEEE Global Communication Conf. (Globecom'06)*, San Francisco, CA, November (**2006**).

47. K. Wang, S. Abu Ayyash, T. D. C. Little, and P. Basu, *Proc. of 2nd Annual IEEE Communications Society Conf. on Sensor and Ad Hoc Communications and Networks (SECON'05)*, Santa Clara, CA, September (**2005**).

48. H. Beigy and M. R. Meybodi, *Advances in Complex Systems* 10, 1 (**2007**).

49. H. Beigy and M. R. Meybodi, *Automatica, Journal of International Federation of Automatic Control* 44 (**2008**), to appear.

50. J. P. G. Sterbenz, R. Krishnan, R. R. Hain, A. W. Jackson, D. Levin, R. Ramanathan, and J. Zao, *ACM Workshop on Wireless Security (WiSe)* 31 (**2002**).

51. C. R. Lin and M. Gerla, *IEEE Journal of Selected Areas in Communications* 1265 (**1997**).

52. W. Chen, J. Hou, and L. Sha, *Proceedings of the 11th IEEE International Conference on Network Protocols* (**2003**), pp. 284–294.

53. D. Liu, *27th International Conference on Distributed Computing Systems* (**2007**).

54. P. Krishna, N. H. Vaidya, M. Chatterjee, and D. K. Pradhan, *ACM SIGCOMM Computer Communication Review* 49 (**1997**).

55. A. Amis, R. Prakash, T. Vuong, and D. T. Huynh, *Proceedings of IEEE INFOCOM* (**2000**), pp. 32–41.

56. O. Younis and S. Fahmy, *IEEE Transactions on Mobile Computing* 366 (**2004**).

57. M. A. L. Thathachar and P. S. Sastry, *IEEE Transaction on Systems, Man, and Cybernetics-Part B: Cybernetics* 32, 711 (**2002**).

58. K. S. Narendra and M. A. L. Thathachar, Learning Automata: An Introduction, Prentice-Hall Inc (**1989**).

59. F. Tobagi and L. Kleinrock, *IEEE Transactions on Communications* 1417 (**1975**).